Assembler Editor Errata

INSIDE FRONT COVER

* The codes listed are BASIC error codes, change them to the the following Assembler error codes.

Error No.	
1	Insufficient memory
2	No start address for DEL command
3	Mini-assembler address error
4	LOAD file error
5	Undefined label reference
6	Error in syntax of statement
7	Label defined more than once
8	Buffer overflow
9	Missing label or * before "="
10	Value greater than 255
11	Invalid null string
12	Incorrect address or address type
13	Phase error
14	Undefined forward reference
15	Line too large
16	Unrecognizable source statement
17	Line number too large
18	Misuse of LOMEM command
19	No starting address
128	[BREAK] key pressed during I/O operation
130	Nonexistent device
132	Invalid command
136	EOF
137	Record longer than 256 characters
138	Device does not respond
139	Device does not return Acknowledge signal
140	Serial bus input framing error
142	Serial bus data frame overrun
143	Serial data checksum error
144	Device done error
145	Read-after-write compare error
146	Function not implemented
162	Disk full
165	Filename error

PAGE vii

* Replace page numbers in contents for:

How to Write Operands	11
Hex Operands	11
Immediate Operands	11
Page Zero Operands	11

Absolute Operands 11
Absolute Indexed Operands 11
Non-Indexed Indirect Operands 12
Indexed Indirect Operands 12
Indirect Indexed Operands 12
Indexed Page Zero Operands 12
String Operands 12

REN Command 16
FIND Command 16

PAGE viii

* In the Appendices, change Appendix 9 title to read:

9 Using the ATARI Assembler Editor
Cartridge to Best Advantage/Example
Programs 63

PAGE 5

* Change Figure 2 caption to read:

Figure 2. Memory map without use of LOMEM.

PAGE 11

* Change the 3rd paragraph to read:

Please refer to the description of the LABEL = directive ...

PAGE 12

* Under Indexed Indirect Operands, add this paragraph:

Using indirect indexed operands will sometimes produce an error 12, although the source code appears to assemble correctly anyway. Use with caution; examine the object code to be certain.

PAGE 17

* Under the REP Command, the first listing in the column should read:

REP/OLD/NEW/

PAGE 18

* In the sample program, add the following line numbers:

```
20
30
40
50
60
70
80
90
```

* In Figure 7, line 50, change the IMY to INY.

PAGE 19

* In the first paragraph under LIST Command, change the 4th line to read:

the printer (#P:), Program Recorder (#C:), and disk drive ...

* Add this paragraph following the 1st paragraph under LIST Command:

The LIST command does not set the display flag, so a LIST containing control characters will execute those functions, instead of printing the characters.

PAGE 20

* Change the program to read as follows, note in particular the indentation and spacing:

```
EDIT
LIST [RETURN]
10   *=$3000
20   LDY #00
30  REP LDX ABSX, Y
40   BNE XEQ SAME PAGE
50   INY TALLY
60   JMP REP
70  ABSX=$3744
80  XEQ=*+$60
90    .END

EDIT
LIST30 [RETURN]

30  REP LDX ABSX, Y

EDIT
LIST 60,80 [RETURN]
```

```
60   JMP REP
70  ABSX=$3744
80  XEQ=*+$60
```

EDIT

PAGE 22

* In the paragraph following ENTER #C:, 3rd line, change it to read:

clears the edit text buffer before retrieving the

* Under LOAD Command format, take out spacing between "device" and "filespec," and insert brackets around both words.

PAGE 25

* In the diagram, change the first line to read:

```
ASM [#D[n]:PROGNAME[.SRC]]
```

PAGE 28

* Add one more space between the line number and pseudo ops on the top half of the page.

* Delete one 0 from the 6th line down so it reads:

```
100   .OPT NOOBJ
```

PAGE 29

* Insert this copy right before the 4th paragraph under Tab Directive:

Where each number corresponds to the number of columns to the right of the line number field to start printing.

Number	Field
1	op code
2	operator
3	comment

* In the last two drawings at the bottom of the page, draw a vertical line to indicate the starting point at number "40."

PAGE 30

* Under BYTE, DBYTE and WORD Directives, change the 2nd line to:

```
200  .BYTE "A,B...N"
```

* In the 4th paragraph from the bottom, the last line, change "Appendix D" to "Appendix 5."

PAGE 31

* In the 4th paragraph down, 2nd line, change it read:

the sample program we used before. Statements 70 and 80 give values...

* Under LABEL= DIRECTIVE, change lines "60" and "70" in the program to "70" and "80."

* In the paragraph following, change the last line referring to "Appendix 4" to " Appendix 5."

* Place this note after the last paragraph of LABEL=DIRECTIVE, and before *=Directive:

Note: The assembler will always assign two-byte values to forward-referenced labels. To use zero page addressing correctly, be certain that all zero page labels are defined ahead of time.

PAGE 32

* In the 5th paragraph down, the first line should read:

The effect of the directive is to reserve 36 locations immediately...

* Under the IF Directive, change the last sentence in the 1st paragraph to read:

If the expression is not equal to zero, all of the code between lines 900 and 990 will not be assembled.

* Rewrite the SOURCE CODE program, note in particular the spacing requirements:

```
0100 ;CONDITIONAL ASSEMBLY EXAMPLE
0120 Z=0
0130  *=$5000
0140  LDA$45
0150  .IF Z@ZNOTEQUAL0
0160  TAX ;THIS CODE ASSEMBLED IFF Z=0
0170 ZNOTEQUAL0
0180  .IF Z-1@ZNOTEQUAL1
0190  ASL A ;THIS CODE ASSEMBLED IFF Z=1
0200 ZNOTEQUAL1
0210  INX ;THIS CODE ALWAYS ASSEMBLED
```

PAGE 35

* In the 5th line from the bottom, the 1st column, change "Tmmmm" to "T or Tmmmm."

PAGE 36

* Under D or Dmmmm Display Memory, change the sentence to read:

Dmmmm, yyyy where yyyy is less than or equal to mmmm shows the contents from mmmm to yyyy, inclusive, wit address "wraparound" occurring at address $FFFF.

* Change the example to:

```
DFFF0,3 [RETURN]

FFF0  68  40  FF FF FF FF FF FF
FFF8  DD  57  B4 E7 77 E4 F3 E6
0000  00  41  41 41
```

PAGE 37

* Delete the sentence that reads "This shows that address 5000..."

* In the 2nd example, the 2nd line, change "500B" to "5008."

PAGE 38

* Change the 1st sentence to read:

The second command puts 31 and 87 in locations 700B and 700E...

* Under Vmmmm Verify Memory, change the 1st sentence to read:

Vmmmm<yyyy,zzzz compares memory yyyy to zzzz with memory...

PAGE 40

* Change the following lines in the program, note in particular the spacing requirements:

```
5001< LDY $1234 [RETURN]
5001  AC3412                 Computer Responds.
< INY [RETURN]
5004 C8                      Computer Responds.
```

* Add this note following the program:

Note: Always leave a space after the < when specifying a mnemonic to be assembled under this command.

* Under Gmmmm Go (Execute Program) and before Tmmmm Trace Operation, add this paragraph:

A BRK (op code = $00) instruction will also stop the GO command. So, even though the debugger does not support an explicit breakpoint facility, you can simulate this facility with a little extra work.

PAGE 41

* Following the example, add this paragraph:

Due to an error in the ROM cartridge, the TRACE function will stop when it encounters a CPY immediate instruction. You should therefore avoid the use of TRACE if your code contains a CPY. If you use S, the STEP function, you can restart the TRACE at the next instruction. However, the status register will not reflect the results of the comparison.

PAGE 47

* Add this definition following the 1st paragraph:

'   The apostrophe indicates that the following character is to be translated into ASCII code.

* Change the program line following the asterisk definition so that the word "HERE" is moved one space to the left closer to the number "50."

* In the last paragraph from the bottom, change "Appendix 6" to "Appendix 5."

PAGE 49

* In the last paragraph on the left side of the page, change the subtitle to read:

Z, PAGE X - Z, PAGE Y - ZERO PAGE INDEXED

* In the 1st paragraph on the right, change the subtitle to read:

ABS, X - ABS, Y - ABSOLUTE INDEXED

* In the last paragraph on the right, change the subtitle to read:

(IND), Y - INDIRECT INDEXED

PAGE 51

* Change line 600 to read:

```
600  LDA #LABEL & $00FF
```

* Change line 620 to read:

```
620  LDA #LABEL/256
```

PAGE 53

* Change the 3rd line in the 1st column to read:

```
.PAGE "MESSAGE"
```

* Change the 5th line in the 1st column to read:

```
.BYTE a,b,...,n
```

* Change the 7th line in the 1st column to read:

```
.DBYTE a, b,...,n
```

* Delete the period in ".LABEL" in the 1st column, 2nd line from the bottom and in the following description.

PAGE 55-60

* Use the ATASCII Character Set from Microsoft BASIC's Appendix K, for Appendix 7, it's more complete.

PAGE 60

* Delete the notes at the bottom of the page.

PAGE 61

* Under ATARI PUBLICATIONS, add these:

ATARI Tech Users Notes	C016555
ATARI OS Listing	CA016557
DOS II Utility Listings	CA016558

* Under OTHER PUBLICATIONS, add these:

The Atari Assembler by Don and Kurt Inman
Reston Publishing Co., Reston, VA

PAGE 63

* Change title to read:

Using the Assembler Cartridge
To Best Advantage/
Example Programs

PAGE 64

* Add this paragraph previous to the 2nd to the last paragraph from the bottom of the page:

Do not assemble directly to tape (ASM,,#C:) except with very short programs. With a longer program it will result in a timeout error. Instead, assemble in memory (ASM) and save to tape (SAVE#C:<startadr,endadr).

PAGE 65

* Change the 2nd sentence in the 1st paragraph to read:

If you have a cassette, type in the following program for loading from cassette tape:

* Then add this program:

```
100 TRAP 300
110 OPEN #1,4,0,"C:":REM open file on cassette for input
120 GET #1,X:GET #1,X:REM throw away first two bytes
130 GET #1,HIBYTE:GET #1,LOBYTE
140 START=HYBYTE*256+LOBYTE:REM starting address
150 GET #1,HIBYTE:GET #1,LOBYTE
160 FINISH=HYBYTE*256+LOBYTE:REM ending address
170 FOR ADDRESS=START TO FINISH
180 GET #1,BYTE
190 POKE ADDRESS,BYTE
200 NEXT ADDRESS
210 GOTO 130:REM check for further sections of code
300 CLOSE #1
```

* Start a new paragraph at "When the machine language program is..."

PAGE 67

* In line 25020, change it to read:

```
25020 FOR I=A TO B:?"[ESC] [ESC]";CHR$(PEEK(I+C));:NEXT I
```

* In the 5th paragraph down, the 3rd line, change it to read:

representation of your object code. To make this line part of your BASIC program

PAGE 68-74

* In the programs on page 68-74, the spacing is wrong, there should be only one space between the 3rd column and the 4th column, one space between the 4th column and the 5th column, and one space between the 5th column and the 6th column.

PAGE 69

* Transpose the information in lines 0250 and 0260 so they read:

```
0250                          PLA
0260                          STA      ATTACK      SET ATTACK TIME
```

* In line 0420, the O in #$BO should be a zero.

* In line 0470, the O in #$OE should be a zero.

PAGE 70

* Add the number 30, directly following line 20.

* In line 0180, change the O in COLORO to a zero.

PAGE 71

* In lines 0630 and 0640, the colons should be semicolons.

PAGE 72

* Under Example 4, lines 10 and 20, the colons should be semicolons.

PAGE 73

* In line 400, delete space from label to read LOOP1.

PAGE 74

* In the 1st line, line 0640, change 8D102 to 8D0102.

* In line 0910, change 8D0AD0 to 8D1AD0.

* Change lines 0940 to 0970 to read:

```
0666   8D18D0   0940           STA   COLPF2
0669   E6CF     0950           INC   DECK       NEXT DECK
066B   68       0960           PLA              RESTORE ACCUMULATOR
066C   40       0970           RTI              DONE
```

PAGE 75

* Add these commands to the beginning of the list of editor commands:

LOMEM bumps the edit text buffer (your source program) upward in memory 7

SIZE gives memory map buffer addresses for edit text buffer and user RAM 6

DOS switches to DOS Menu, destroying current assembler RAM program in process. (No page reference in this manual.)

* Change the 2nd to the last editor command to read:

SAVE #C:<xxxx, yyyy

PAGE 77

* In the program, move *=$600 to the 2nd column above LDA.

* In the same program, change END to .END and move to the 2nd column below STA.